www.bsc.es

Urbana, IL, 17th July 2017

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

EXCELENCIA SEVERO OCHOA

Online diagnostics generation in high resolution climate simulations

JLESC 2017

Kim Serradell Maronda

Outline

- Earth activities at BSC
- EC-Earth climate model
  - More resolution, bigger outputs
  - Compute diagnostics
  - Adaptive ensemble workflow
- Discussion
  - Earth sciences in JLESC

Barcelona Supercomputing Center
Centro Nacional de Supercomputación
EXCELENCIA SEVERO OCHOA

BSC Earth Department

BSC Earth Sciences Department

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

EXCELENCIA SEVERO OCHOA

What

Environmental forecasting

Why

Our strength …
- research
- operations
- services
- more than 60 people working together

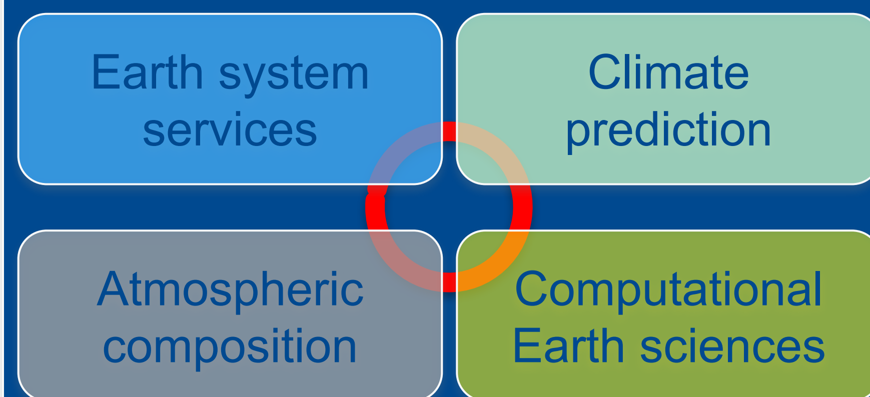
How

Develop a capability to model air quality processes from urban to global and the impacts on weather, health and ecosystems

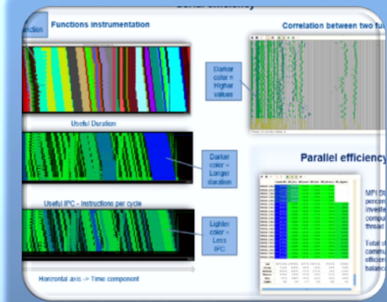
Implement a climate prediction system for subseasonal-to-decadal climate prediction

Develop user-oriented services that favour both technology transfer and adaptation

Use cutting-edge HPC and Big Data technologies for the efficiency and user-friendliness of Earth system models

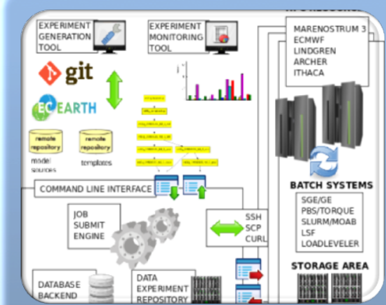


Computational Earth Sciences



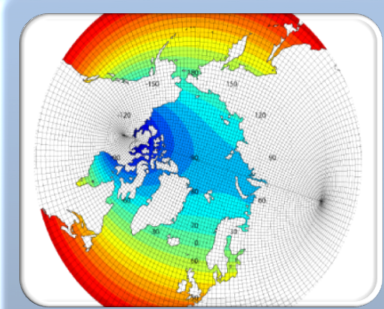
Performance Team

- Provide HPC Services
- Apply new computational methods



Models and Workflows Team

- Development of HPC user-friendly software framework
- Support the development of climate and atmospheric research software



Data and Diagnostics Team

- Big Data in Earth Sciences
- Provision of internal and external data services
- Visualization

www.bsc.es

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

EXCELENCIA SEVERO OCHOA

Let’s talk about applications

EC-Earth

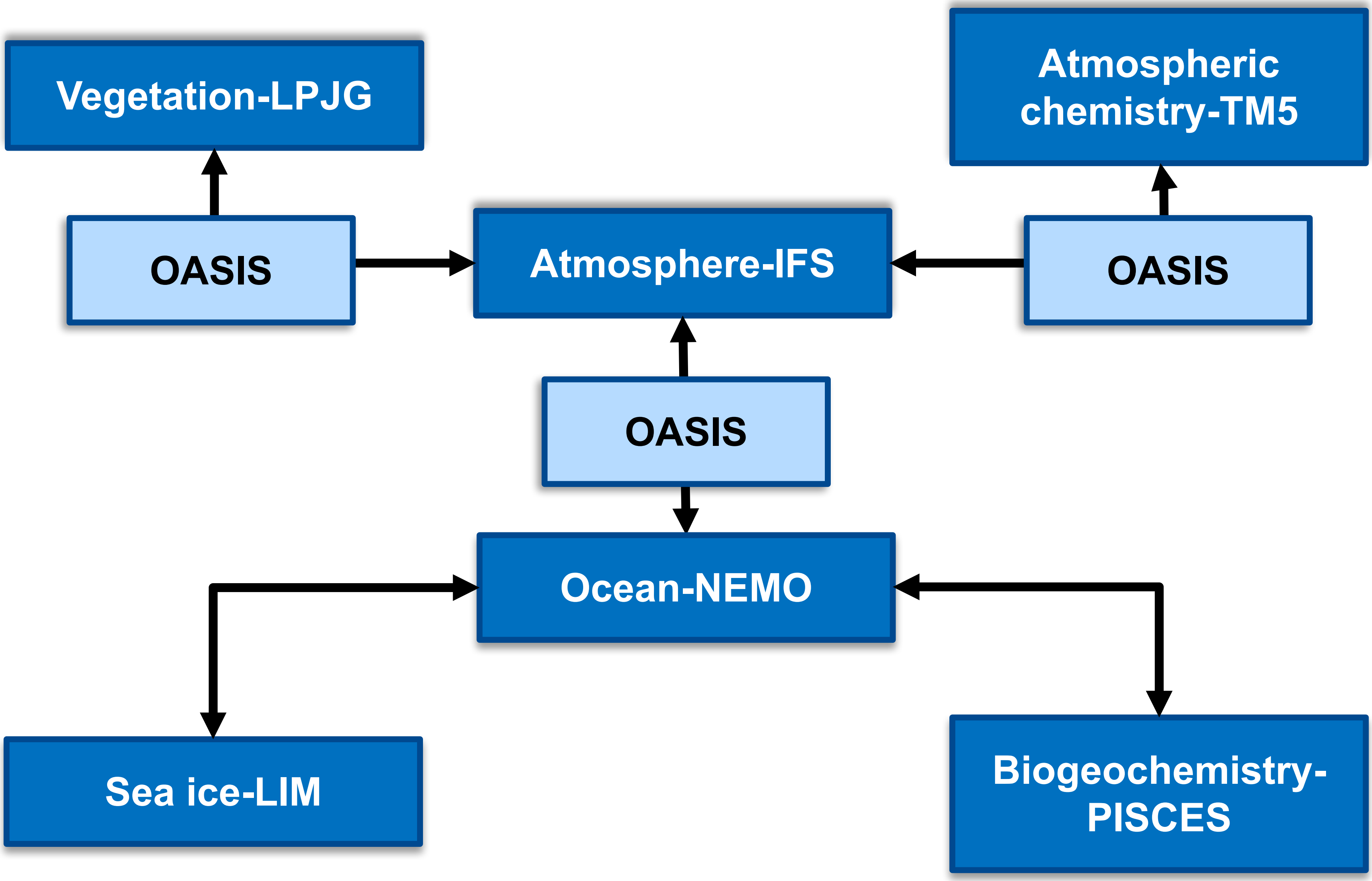
- EC-Earth climate model
  - IFS for atmosphere (ECMWF)
  - NEMO (IPSL) for ocean

- EC-Earth is developed as part of a Europe-wide consortium thus promoting international cooperation

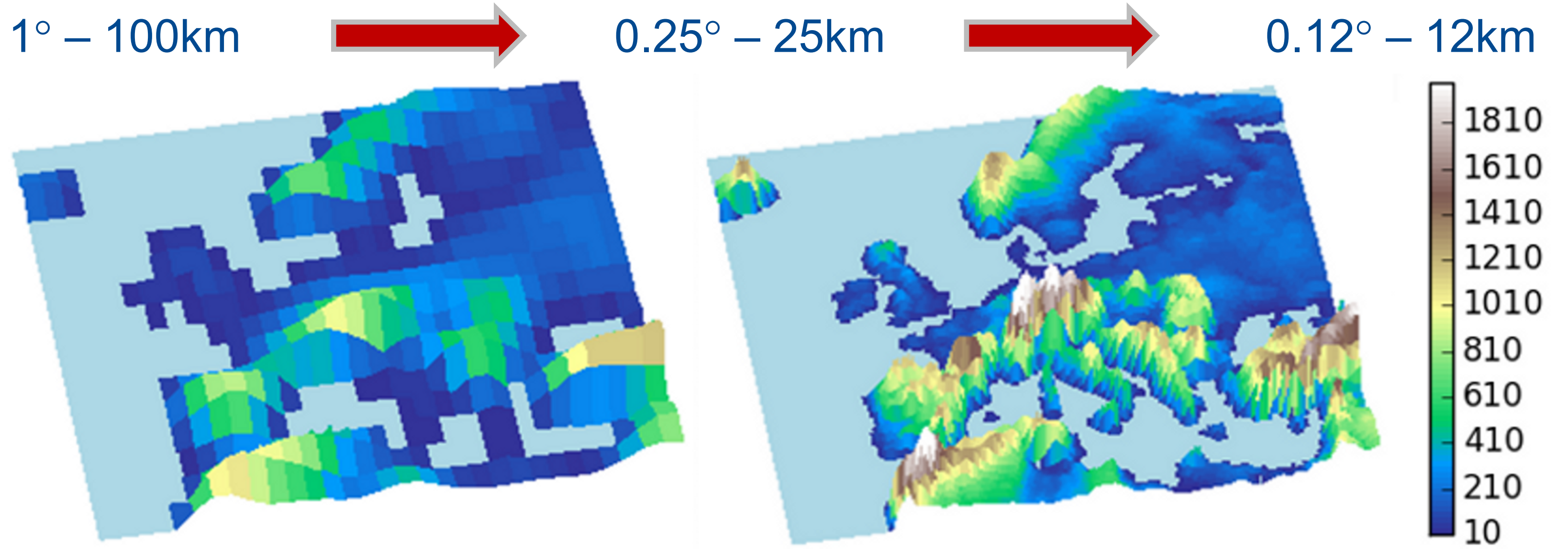
- Coupled online with OASIS y XIOS as I/O server

- About JLESC collaboration:
  - EC-Earth deployed in MIRA (Argonne) with an INCITE call
  - Currently in K computer (in progress)

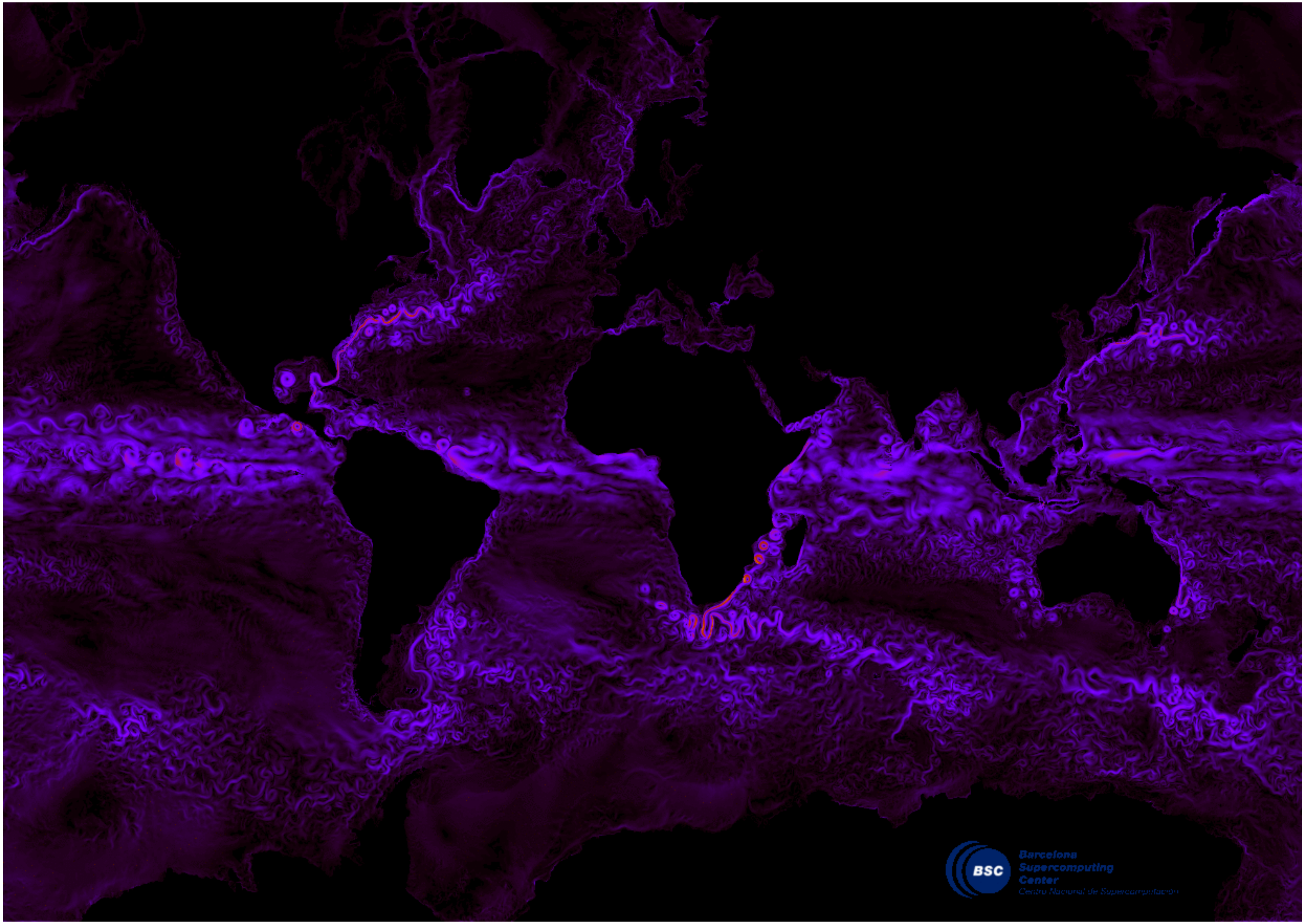
The Earth system model: EC-Earth



More resolution, bigger outputs

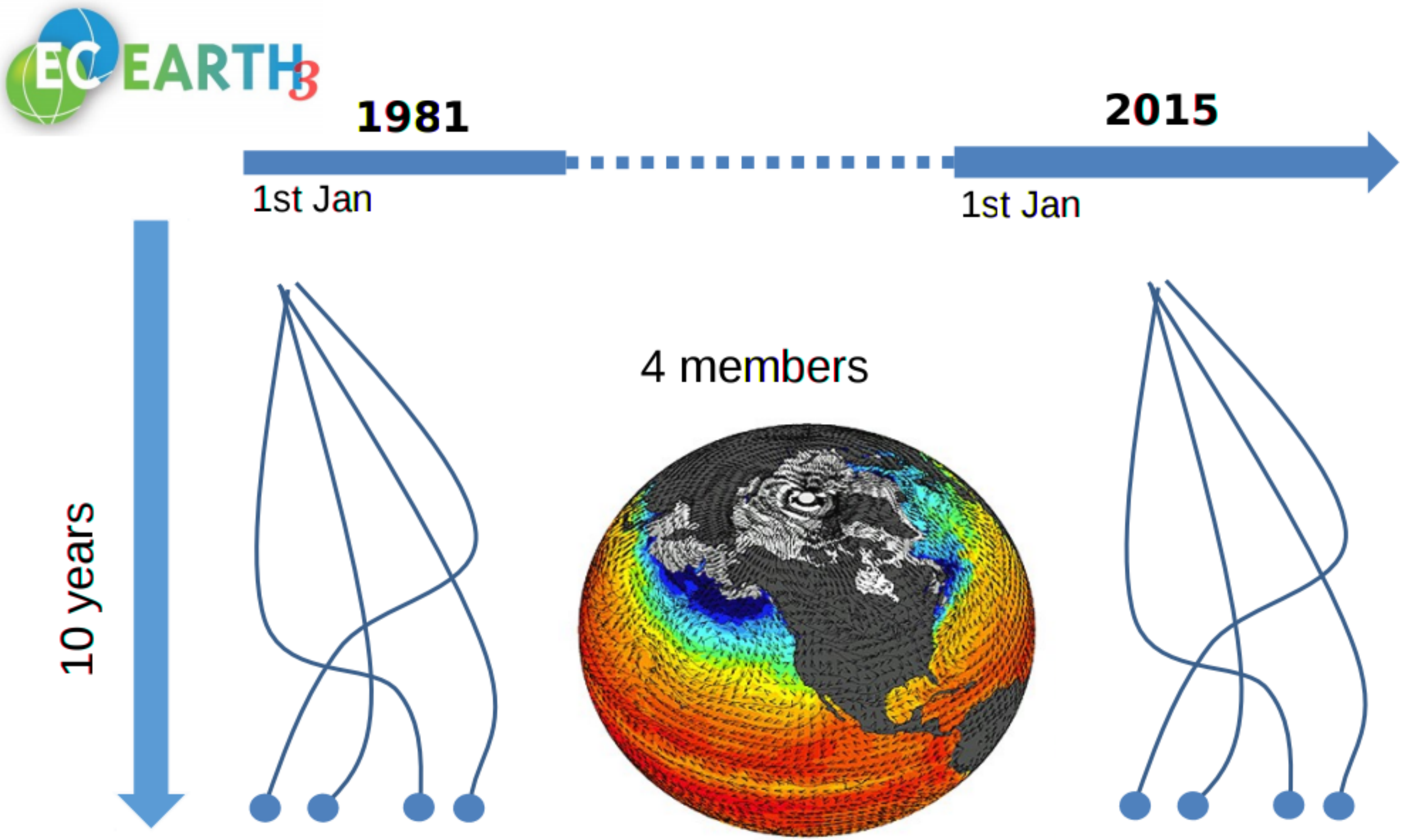


EC-Earth	Horizontal resolution (atmosphere/ocean)	Output size per year (in NetCDF4, restarts not included)
standard resolution	T255/ORCA1 60km/100km	26 Gb
high resolution	T511/ORCA025 40km/25km	170 Gb
very high resolution	T1279/ORCA012 25km/12km	1 Tb



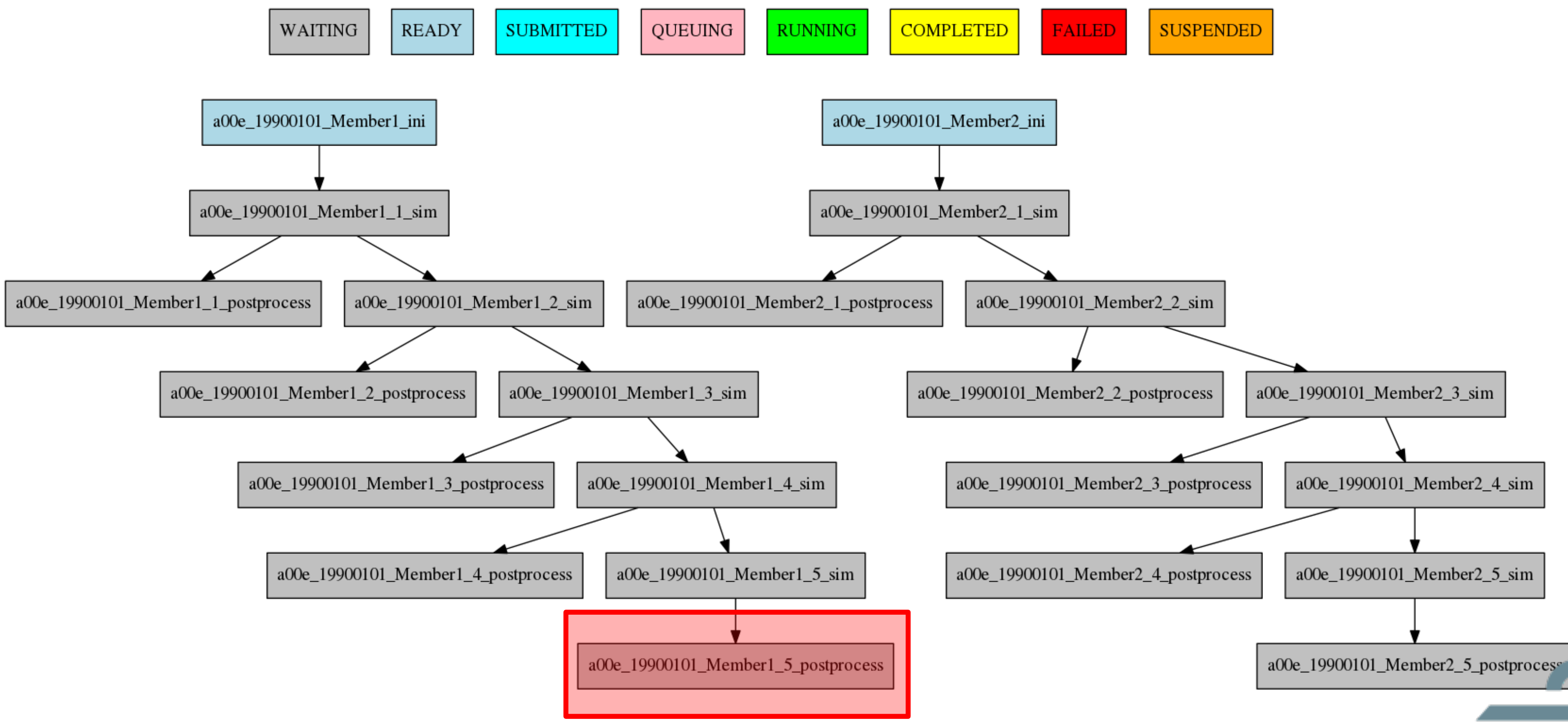
Ocean currents T1279-ORCA12 EC-Earth run (courtesy L. Brodeau)

Running a decadal experiment



Autosubmit

- Autosubmit
  - A versatile tool to manage Weather and Climate Experiments in diverse Supercomputing Environments
  - https://pypi.python.org/pypi/autosubmit



D. Manubens (BSC)

Computing diagnostics today

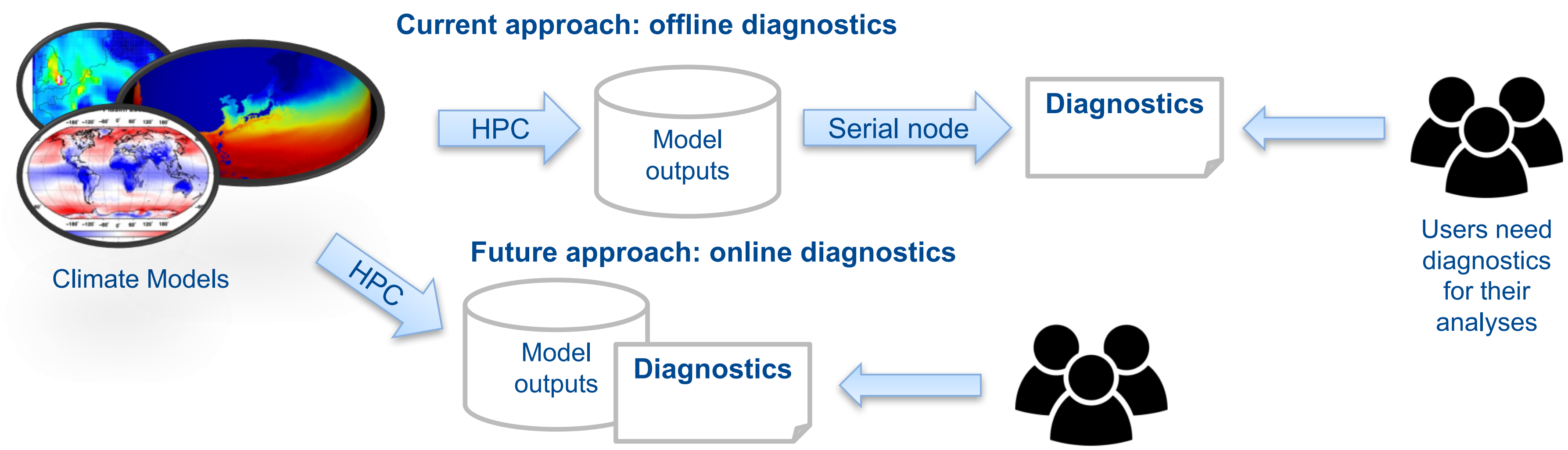
- Diagnostics are computed after simulation
  - Only a subset of variables is saved

- Diagnostics generation is mainly sequential

- Different languages and technologies:
  - Fortran codes
  - Bash scripts using tools like CDO or NCO
  - Python scripts using modules like numpy, iris, pycdo…

- We use fat nodes with a big amount of memory to compute these diagnostics.

Online diagnostics for climate models



- Diagnostics computed as Analytics as a Service
  - Diagnostics online (during model run)
  - Reduced data traffic
  - Diagnostics possible on the computing nodes
  - New diagnostics (data mining of extremes) possible
  - The user gets the results faster → crucial to adapt to climate change and to develop climate services (public and private)

XIOS to create diagnostics

- I/O server developed at IPSL (FR)
- Writes outputs in NetCDF
- Configuration using XML files (flexible)

- Right now, only NEMO (ocean) uses XIOS
- BSC is porting XIOS to IFS (atmosphere)

- XIOS computes diagnostics but some of them are computationally expensive

Porting Earth diagnostics to GPUs

- How is defined the ocean heat content?

$$heatc = \rho c_p \int_{h_2}^{h_1} \theta_o$$

- It is the integral of the seawater potential temperature between two levels multiplied by the density and the specific heat of water

- Using python 2.7.9 and PyCuda package
  - Python wrapper for Nvidia CUDA
    - https://pypi.python.org/pypi/pycuda

Snippet

```
    compute_weight = ElementwiseKernel(
        weight_declaration,
        weight_kernel,
        "compute_weight")
    gpu_weight = tuple(pycuda.gpuarray.empty_like(gpu_mask) for x in range(len(layers)))
    args = (gpu_e3t, gpu_depth, gpu_mask) + gpu_weight
    compute_weight(*args)
    del gpu_e3t
    del gpu_depth
    del gpu_mask
    thetao = iris.load_cube(DATA_FILE, 'sea_water_potential_temperature')
    gpu_thetao = pycuda.gpuarray.to_gpu_async(thetao.data.astype(np.float32))
    shape = gpu_thetao.shape

    block_size = 256
    grid_size = ((shape[3] * shape[2]) // block_size) + 1
    mod = """
     __global__ void ohc(float *thetao, float *weigth, float* ohc)
     {{
       int iohc = blockIdx.y * {0[2]} * {0[3]} + blockIdx.x * {1} + threadIdx.x;
       float temp=0;
       if (threadIdx.x + blockIdx.x * {1} >= {0[2]}*{0[3]}){{
           return;
       }}
       for(int lev=0; lev < {0[1]}; lev++) {{
           int idx = blockIdx.y * ({0[1]}*{0[2]}*{0[3]}) + lev * ({0[2]}*{0[3]}) + blockIdx.x * {1} + threadIdx.x;
           temp += thetao[idx] * weigth[idx];
       }}
       ohc[iohc] = temp;
     }}
     """.format(gpu_thetao.shape, block_size, grid_size)
```

Test case

- Ec-Earth 3.2 ORCA 0.25 L75
  - 1442 x 1050 x 75 grid points
  - Irregular across all dimensions (although we are only concerned by the z dimension)
  - One timestep
- Test machine: CTE-POWER
  - 1 login node and 6 compute nodes, each of them:
    - 2x IBM PowerNV 8335-GTB @ 4.00GHz (10 cores and 8 threads/core, total 160 threads per node)
    - 256 GB of main memory distributed in 32 dimms x 8GB @ 2400MHz
    - 2x nVidia Pascal P100 GPU with 16GB of memory.
    - GPFS via two fiber links 10 Gbit
  - Test run at the login node
- Calculate ocean heat content from 0 to 300 meters depth

Performance

- CPU mean time execution: 71.68 seconds
- GPU mean time execution: 13.92 seconds
  - Actual time spend in GPU related tasks:
    - ~ 200 ms moving data (4*450Mb aprox.) from and to GPU memory (taking advantage of NVLink)
    - ~ 5 ms computing on the GPU
- Speed up of more than 5x
- When computing more than one timestep, you can overlap computation with the following

- Next test, porting to FPGAs using PyOpenCL (same developer as PyCuda
- Sea ice area and extend has also been ported to GPUs
- Running simulations in an heterogeneous cluster

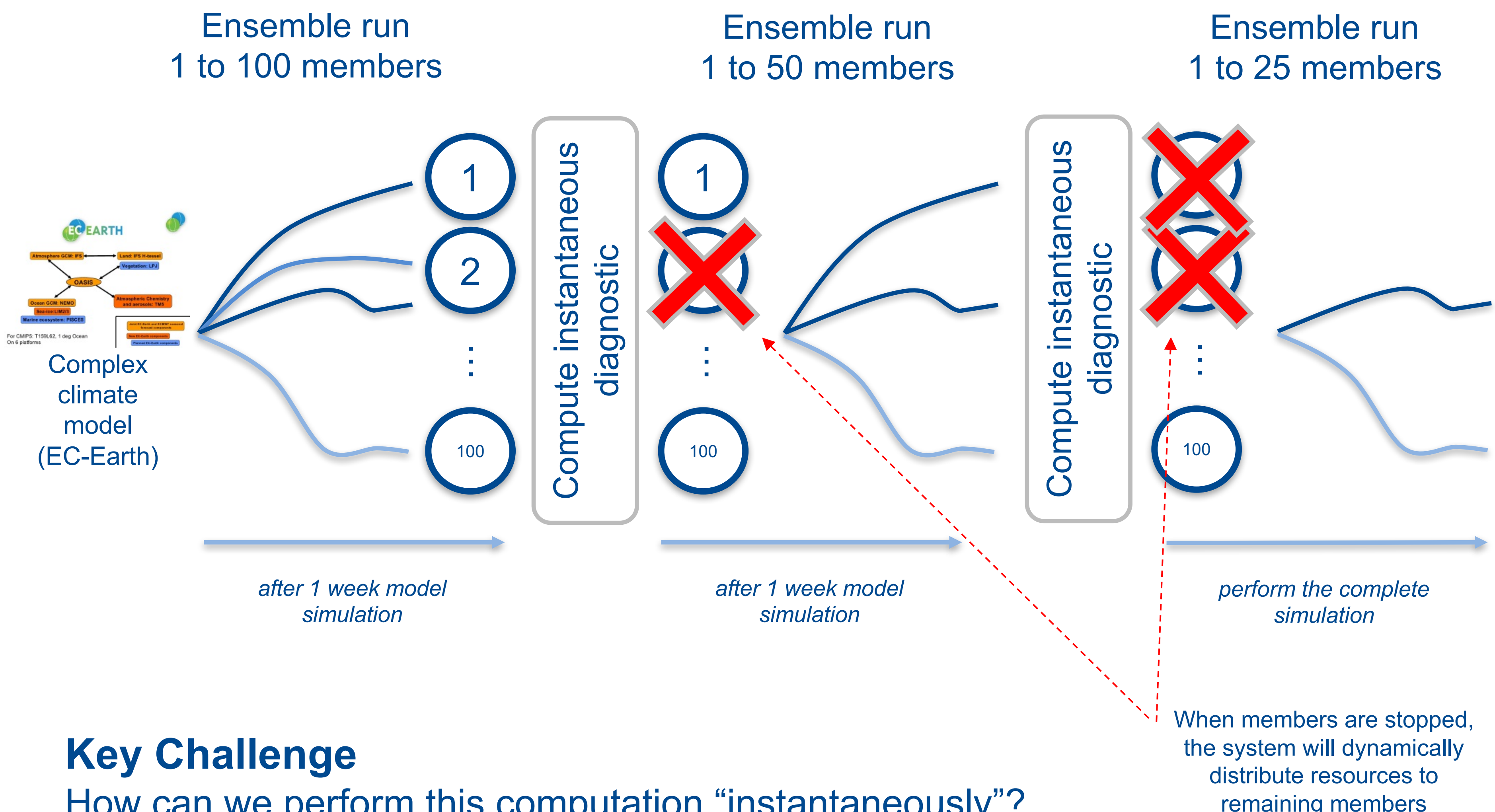
www.bsc.es

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

EXCELENCIA SEVERO OCHOA

Going further

Adaptative ensemble simulations



Key Challenge

How can we perform this computation "instantaneously"?

- Select a right diagnostic (climate science)
- Work in XIOS I/O server and disk technologies (computer science)
- Using PyCompSs to orchestrate the workflow

www.bsc.es

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

EXCELENCIA SEVERO OCHOA

Discussion

Discussion: Post process in JLESC

- Experiences in post processing Earth System model outputs in JLESC?
- How do you produce and analyze unusually large model output efficiently?
  - Global simulation of nearly 1km from Riken.
- Share experiences.
- Tools, performance and diagnostic selection are key topics to achieve successfully the generation and post-process of such simulations.

Questions

www.bsc.es

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

EXCELENCIA SEVERO OCHOA

Thank you!

For further information please contact
kim.serradell@bsc.es